UNITED STATES PATENT APPLICATION

OF

MICHAEL OLENICK
Costa Mesa, California, and

JEFFREY JOOSTE
Gig Harbor, Washington

FOR AN INVENTION ENTITLED

SYSTEM AND METHOD FOR GENERATING CUSTOMIZED AND/OR PERSONALIZED DOCUMENTS

DOCKET NO. 310048-511199
SHEETS OF DRAWINGS: 11

Oppenheimer, Wolff & Donnelly LLP
2029 Century Park East, 38th Floor
Los Angeles, California 90067

I. RELATED APPLICATIONS

The present patent application claims the benefit of U.S. Provisional Patent Application No. 60/201,234, which was filed on May 1, 2000 and is entitled "System and Method For Generating Customized and/or Personalized Documents."

II. BACKGROUND

A. Field of the Invention

The present invention relates to designing and printing documents and, in particular, to a system and method for designing a document online and then printing the document at a printer.

B. Prior Art

Computer software for custom document design enables users to design and print custom-created documents, such as invitations, business cards, posters, and the like. There is wide-spread demand for custom document design software. Retailers need custom-printed point-of-sale displays and advertising materials; direct marketers utilize printed materials for direct mailings and coupons; promotion firms utilize printed materials to announce events.

Until recently, custom document design software was typically resident on a user's personal computer. Software such as the highly-regarded AveryWizard® and LabelPro® programs allow the user to customize and personalize documents on a personal computer, and then print the documents at a local printer. While these programs are very popular and are in wide-spread use, the development of the Internet opens up new possibilities for custom document design. With existing PC-based document customization and personalization

software, the user must take time to install the software on the personal computer. Once the software is loaded onto the personal computer, it must be periodically upgraded, which takes further time.

There are now a few client-server, network-based custom-printing systems that allow a user to access software that is stored on a server, design a document while connected to the server, and then print the document after the design has been completed. However, existing network-based printing systems do not provide sufficient speed and flexibility. One common approach is to first build a graphical representation of a document to be printed (or "render" the document image) on the server. Then, the graphical representation is converted into a format such as a Portable Document Format ("PDF"), and the PDF file is transmitted to the client. The client then renders the image again on the client before printing it out. The process of rendering the image two times – once at the server and again at the client – is slow and inefficient. The capacity of the server to serve multiple simultaneous users is also limited by the need to undertake the computationally-intensive task of rendering the graphic on the server.

Another approach is to start with a page description file in a format such as PDF. The initial page description file describes an initial, uncustomized layout for a document. The PDF file is then modified to include changes that the user has specified, such as inserting the user's name into the document, changing the font size, and so on. This approach is awkward because, among other things, modifying an existing PDF file can become very complicated. The approach is prone to error, is difficult for back-end programmers to implement, and is difficult to modify once it is implemented.

Another problem with existing network-based systems for designing and printing customized documents relates to the manner in which the web browser communicates with the server. A web browser is a mechanism for housing software and viewing text and graphics from the Internet. The browser running on the user's computer accepts the data from the user and formats the data into a data stream using HyperText Markup Language ("HTML"). The phrase "data stream," when used herein, applies equally to data received as if it were an incoming stream and to data that has been received in this manner and stored in a file. HTML is a standardized notation for displaying text and graphics on a computer display screen, as well as providing more complex information such as animated video and sound.

Newer browsers are equipped to format data using Extensive Markup Language ("XML"). XML is designed to provide greater flexibility than HTML. While HTML uses a predefined set of elements to markup documents, XML utilizes a formal syntax for describing relationships between entities, elements and attributes that make up XML documents. XML represents a significant upgrade over HMTL, but many browsers do not support XML.

III. SUMMARY OF THE INVENTION

The present invention relates to an efficient method and system for generating customized and/or personalized documents, typically over a network having a client and a server.

In particular, one embodiment of the present invention is an efficient method for generating and printing customized documents in a system having a client communicable with a network and a server communicable with the network. An interactive form is displayed on the client. User information is entered onto the interactive form, and is transmitted from the client to

the server over the network. Default document parameters are obtained from a template file. Instructions to a page description file builder are formulated based upon the default document parameters and the user-defined information. A page description file is built based upon the instructions, and the page description file is transmitted to the client. The page description file is rendered for the first time at the client.

The method may also include various features and steps. The step of obtaining default document parameters from a template file may include parsing the template file. The template files may be in the form of statements, such as Extensible Markup Language (XML) statements. The step of transmitting user-defined information comprises transmitting information in Hypertext Markup Language (HTML) code, with an option value format having a syntax comprising a token, a directive and a parameter. This structure may be referred to as "pseudo-XML" because it mimics XML functionality. Pseudo-XML is also extensible in that the set of parameters may be extended indefinitely, yet the pseudo-XML is backwardly compatible with HTML browsers. The user information will typically include a variety of information, such as name, address, telephone number, facsimile number, e-mail address, text message, selection of a pre-defined graphic, and/or type of document to be generated. The user information may also include other information such as font type, font color, font size, location of text on the document, and/or location of graphics on the document.

The method may further include the step of printing the rendered page description file on a client-controlled printer. A table of printer driver characteristics may be stored on the server. The table may be created by a single source, or may be built dynamically based on data provided by online users. Using information about the printer characteristics, the page description file is built to compensate for any tendency in the printer driver to print the ultimate page in a position

on the page other than that desired. That is, the PDF file is built to compensate for any tendency of a particular printer driver to print the image at an offset to where the image should be on the page.

The client-controlled printer can be a standard office printer, such as a laser printer, inkjet printer, or bubblejet printer, or a commercial printer. Alternatively, the page description file may be transmitted from the client to a separate printing facility for printing on a commercial printer.

The page description file can be a Portable Document Format (PDF) file, a Postscript file or another format known in the art or developed in the future. The client may be a desktop computer, a Personal Digital Assistant (pda), or another type of client that can interact with a server over a network, including a telephone that has a browser. The network can be an intranet or the Internet, or the means for connecting the client to the server can be a direct connection without the use of a network at all.

The step of formulating instructions can include formulating instructions in accordance with an application programming interface (API). A java servlet or other computer program can be used to perform the step of formulating instructions. A Hypertext Markup Language (HTML) browser on the client can perform the step of transmitting the user-defined information from the client to the server, although as browsers evolve it is expected that the browser will be compatible with Extensible Markup Language (XML), which is more versatile than HTML.

The default document templates may be generated with a Graphical User Interface (GUI), and stored on the server. A graphical visual representation of the template may be created with the GUI and then saved as an output data file. The templates may include default values for at least one of the following: font type, font color, font size, background color, location of text on

the document, location of graphics on the document, size of the document, and/or shape of the document.

The method may also include the step of determining characteristics of a client-controlled printer. The page description file can then be built for compatibility with the particular client-controlled printer. The quality and/or printed appearance of the printed graphic is thereby enhanced. For example, the page description file can be built to ensure that the document is printed at a particular page location consistently from printer to printer.

In accordance with another aspect of the invention, an efficient method for generating customized electronic documents for printing includes receiving via an online connection an identification of a document template. A document template defining default attributes of a document to be printed is then obtained. Document customization information and document personalization information is then received, and a set of instructions to a page description file builder is generated. The instructions instruct the page description file builder to build a document based upon a combination of the default attributes of the document to be printed, the document customization information, and the document personalization information. A page description file is built from the set of instructions, and is transmitted to a client for rendering and then printing at a printer. For efficiency, the steps of formulating a set of instructions and building a page description file are accomplished without first graphically rendering the document, and the document is graphically rendered for the first time at the client.

In another aspect of the invention, a processing server is programmed to receive information via a network from a user. The information can include document customization information and/or document personalization information. The server is also programmed to

obtain a document template that defines default attributes of a document to be printed, and to formulate a set of instructions to a page description file builder. The instructions instruct the page description file builder to build a document based upon a combination of said default attributes of the document to be printed, the document customization information, and the document personalization information. The server is programmed to build a page description from the set of instructions with the page description file builder, and to transmit the page description file to a recipient. The server is programmed to build the page description file without graphically rendering the document. The processing server may further be programmed to determine characteristics of a printer onto which the rendered document will be printed, and to customize the page description file for the printer as, for example, to assure that the printer prints the rendered document in the proper location on the sheet.

It should be noted that pseudo-XML itself, as introduced above and discussed in more detail below, is considered to be an invention. That is, the concept of an extensible psuedo-XML code having a syntax that includes a token, a directive and a parameter, and that is compatible with HTML-based browsers, is itself an invention. Pseudo-XML can be utilized in a variety of applications, including but not limited to customization and/or personalization of documents, graphic files, and the like. One advantage of psuedo-XML is that it simplifies the creation and maintenance of data entry screens, as compared to hard-coding such screens. In one embodiment of psuedo-XML, programmers may make substantive changes to the user interface (UI) using HTML code rather than more complicated types of code.

The present invention also extends to particular models of doing business with the technology. Various other aspects, features and embodiments of the invention will become apparent in the following Detailed Description, in the drawings, and in the Claims.

IV. BRIEF DESCRIPTION OF THE DRAWINGS

Fig. 1 is a diagram of a networked computing environment having a server programmed to generate graphic document files in accordance with customization and/or personalization data entered at a client;

Fig. 2 is a block diagram of one embodiment of a method for generating and printing documents over a network;

Fig. 3 illustrates one embodiment of an interactive form that permits the user to select from a variety of different document types;

Fig. 4 is an interactive form for inputting user customization and personalization information;

Fig. 5 is an interactive form that displays a preview version of a customized and personalized document and that permits the user to enter revised customization and personalization data, as desired;

Fig. 6 is a screen that displays an image of the rendered document to be printed, the document having been rendered at the client but not at the server;

Fig. 7 illustrates one embodiment of a final, printed document, with lines of perforation separating individual business cards;

Fig. 8 is a block diagram of an alternative embodiment of a method according to the present invention, in which the step of previewing the document prior to generating the page description file has been omitted;

Fig. 9 is a block diagram illustrating a method for generating the page description file on the server;

Fig. 10 is a block diagram of a method for generating default document templates with a Graphical User Interface (GUI); and

Fig. 11 is a block diagram of a method of generating and printing a customized and/or personalized document in which the customization and/or personalization information is transmitted from a server on which it is stored to a second server that generates a page description file.

V. DETAILED DESCRIPTION OF A PREFERRED EMBODIMENT

Figure 1 illustrates a client-server model in which a client computer 10 connects to a server computer 12. The client computer 10 is connected to the server 12 via a LAN (local area network), a phone line, or a TCP/IP based WAN (wide area network) on the internet. A client/server network set-up enables many clients to access the same applications and files that are stored on the server 12. The client 10 is typically a personal computer, which has various peripheral devices such as a keyboard 14, a monitor 16, a mouse 18, and a floppy disk drive 20. The client 10 is also typically connected to a printer 22, which may be an ink jet printer, a laser printer, any of a variety of different digital printers, a commercial printer and various other types of printers that may be connected to a client.

The client 10 need not be a personal desk top computer as shown in Figure 1, but may instead be a Personal Digital Assistant (PDA), an advanced wireless phone, or other device capable of connecting to a server 12. Whatever specific form the client may take, a common characteristic is that the client has a browser, which allows the user to read hypertext and act as the client to the server 12. Specific examples of suitable browsers include Microsoft's® Internet Explorer and Netscape® Navigator. The browser may be embedded into a consumer electronic

device, such as a telephone, a PDA, electronics in an automobile that interact with a network via a wireless modem or other wireless device, or any of a wide variety of devices that can communicate with another device across a network.

Servers such as server 12 are well known in the art. They are typically powerful PC's or other types of powerful computers that are programmed with application software for processing information received from various clients. Server 12 typically has memory in which web pages and other information is stored. In the case of the present invention, the server is programmed with software for receiving information from the user and generating a page description file in response to the instructions from the user. Server 12 includes various web pages that are stored in the memory of the server. Many of these web pages are interactive forms that are transmitted to the client. The user views the form on the client and enters information onto the form. The browser then transmits the user-entered information to the server.

The user may input the information into the interactive form using the mouse 18, the keyboard 14, with voice commands, or various other means for inputting information into an interactive form that are known in the art. The server 12 is typically programmed with software, such as a Java servlet or other computer program, to process the information that the server receives from the client.

An important function of the presently preferred embodiment of the invention is to allow the user to customize and/or personalize documents such as business cards, greeting cards, notes and other printed types of media. The system takes the customization and personalization information that the user has input and generates customized documents in accordance with the customization and personalization information. Figure 2 illustrates the process of creating a

customized and/or personalized document from the user's perspective. The user begins the process by accessing the first web page from the server. This step is identified as Step 100 in Figure 2. At Step 102, the server transmits a web page corresponding to an interactive form, which is displayed on a display device at the client 102. The interactive form may include a list of different document types from which the user may choose by clicking on a particular document type. For example, in Figure 3, a list 30 of various different types may be displayed on the monitor of a client in Step 102. In the particular example of Figure 3, the user may choose from a variety of different document types, such as business cards, tent cards, greeting cards, calendars, labels, index tabs, CD-ROM labels, ZIP disk labels, and other types of standard documents.

At Step 104, the user clicks on one of the various document types listed in the interactive form of Figure 3, to specify the type of document that the user wishes to generate. For example, if the user wishes to design business cards, the user would click on the words "business cards" 32 on the interactive form of Figure 3. Figure 3 illustrates just one type of interactive form, and there are many different types of interactive forms that may be used. These various types of interactive forms are well known in the art, and include such features as blanks into which the user may type a particular desired type of document, pull-down menus, buttons on which the user can click with a mouse, and other standard web page features.

Once the user has selected a particular document type in Step 104, the server 12 transmit a second interactive form to the client. The second interactive form asks the user to enter particular information appropriate to the document type that the user has selected. When the interactive form is displayed on the client (Step 106) the user then enters particular information

onto the second interactive form in order to customize and personalize the document. This step is illustrated at Step 108 in Figure 2.

Figure 4 illustrates one embodiment of a second interactive form 36. The form 36 provides various opportunities for the user to customize the document to be generated. For example, the user may select from a variety of different business card layouts 38A-C. The user may select the type style, or font, that is to be used to print their name at menu 40, and may specify the type size, type color and justification of the name at pull down menus 42, 44 and 46. The user may also adjust the location of the name on the card nudging it up, down, to the right or left by selected increments from the default position defined in the default template. For example, pull down menus 48 and 50 allow the user to nudge the name up, down, right and/or left at specified increments, as desired. The second interactive form 36 also provides the user with an opportunity to customize the title and address block of the business card with pull down menus 52 and 54, respectively. The user may define the particular graphic to be placed on the card, either by entering a file location at input block 56, or by selecting a predefined graphic from among a graphic menu 58.

It should be noted that in some embodiments of the invention, restrictions will be placed on what document defaults the user will be permitted to override. So, for example, in some embodiments the user will be permitted to change the font, but not the font size or color. Which particular default parameters the user will be permitted to change will depend on the desired characteristics of the system for a particular application or type of document. The invention is quite flexible in this regard, and encompasses systems that provide the user with a wide range of customization and personalization options, to systems that provide the user with no such options

at all, as when another computer transmits data to the server for formulation into a page description file without the user providing customization and/or personalization information to the server.

Returning to the embodiment of Fig. 4, the interactive form 36 also permits the user to personalize the document. That is, the user is permitted to enter personal information, such as a name, at input window 60, a company name at input window 62 and address, telephone, and e-mail information at input block 64. Between the customization information and the personal information, the user is able to control numerous parameters defining the ultimate format and appearance of the business card.

At Step 110, the user has submitted the customization and personalization information as illustrated in Figure 4 to the server. In the specific embodiment of an interactive form that is illustrated in Fig. 4, the user submits the information she or he has entered by clicking on the "Preview" button. Upon receiving the customization and personalization information, the server then generates a preview version of the document 66, as illustrated in Figure 5. In one particular embodiment of the invention, the preview version 66 of the document to be printed is a quickly generated graphical image that may be in a relatively low-resolution format, such as a low-resolution Joint Photographic Experts Group (JPEG) format, progressive JPEG, GIF, interlaced GIF or other formats.

It should be mentioned that a relatively low-resolution image is presently preferred for the preview step because a low-resolution image file can be transmitted quickly across current networks. However, on high-speed networks with high-speed servers and significant band width, the preview image need not be a low-resolution image. That is, it is possible to

implement a preview step that presents a high-resolution preview image to the user, within the scope of the invention.

After viewing the preview version 66 of the image, the user may wish to modify certain aspects of the document. Consequently, the user is allowed to reenter customization and/or personalization information, as desired, on the second interactive form, which is numbered 36' on the form in Figure 5. The user may reenter the customization and/or personal information as often as he or she likes, and may preview the revised version over and over again as changes are made. When the user is satisfied with the preview image 66 of the document, the user clicks on the final step button 70 to indicate to the server that the user is satisfied with the customization and personalization information. The software on the server then generates a high quality final version of the document to be printed. This high quality version of the document to be printed may be a PDF file, which is transmitted from the server to the client at Step 114.

When the client has received the page description file, the client then renders a graphic based on the page description file at the client. This step of rendering the page description file 116 may be accomplished with standard graphic rendering software, such as Acrobat Reader®. Alternatively, another standard graphic rendering software that is compatible with the type of file that the server has generated may be used.

In one embodiment of the invention, the page description file is a Portable Document Format (PDF) file, but it can alternatively be in other formats, such as versions of Encapsulated Postscript (EPS), or a variety of other formats. As used herein, the term "page description file" preferably refers to a file, such as a PDF file, that describes a page in terms of text, embedded bitmap images, and/or the like. A page description file is different than bit-mapped, raster image

file formats such as a BMP, Graphics Interchange Format (GIF), Tag Image File Format (TIFF), or PCX. As opposed to those bit-mapped file formats, which result in relatively large file sizes, a page description file instead provides a graphic rendering program (such as Adobe® Acrobat Reader®) with the information needed to render a graphic, typically without providing specific information for every individual bit or pixel of a display space. Consequently, a page description file is usually more compact than a bit-mapped file, and is better suited for transmission over a network.

Once the client has graphically rendered the document at Step 116 using the page description file, the user then has the option at Step 118 of printing the rendered document at a client controlled printer, such as printer 22 in Figure 1. As an alternative to Step 118, the user may save the file to be printed later.

Returning to Step 116, Figure 6 illustrates a graphic image 70 that has been generated with the Acrobat Reader® software on a terminal screen 72. As seen in Figure 6, the rendered document image 70 corresponds to what will be a sheet of printed business cards. In an embodiment of the invention in which the printer to be used is directly connected to the client, or is connected to the client through a network such as a LAN (local area network), the Acrobat Reader® software or other software being used to render the graphic, may print the image 70 directly to a printer.

Figure 7 illustrates an example of a final printed sheet of business cards 74 that has been imprinted with the image 70 of Figure 6. In the particular embodiment of Figure 7, the sheet 74 includes various lines of perforations 76 and 78 that separate the individual business cards from

one another. The lines of perforation 76 and 78 may be lines of micro-perforations, which are very fine and closely spaced perforations that permit the user to separate the individual cards without leaving behind a ragged edge where the perforations had been. Such types of business card sheets are well known in the art, for example, as Avery Dennison Corporation Product No. 5376.

It should be understood that the present method of generating and printing customized and personalized documents may be accomplished with various additional steps, or with fewer steps. For example, in Figure 8, the step of previewing the graphic before final generation and printing of the graphic has been omitted. That is, once the user has entered his or her customization and personalization data, the user does not necessarily need to preview what the document will look like prior to the final page description file being constructed. Thus, for example, Figure 8 illustrates a sequence of steps in which the preview steps have been omitted. The steps in Figure 8 are typically the same steps as in Figure 2, although the steps are numbered with a prime (" ' ") to indicate that they are part of an alternative embodiment of the method.

To this point, our discussion of the present invention has been directed primarily toward the perspective of the end user, who uses the client to define and ultimately print customized and personalized documents. However, an important aspect of one embodiment of the present invention is that the system generates the final page description file in a particularly efficient manner. That is, as compared to prior art methods in which the server itself graphically renders the document prior to transforming the graphic into a page description file to be sent to the client, the preferred embodiment of the present invention never renders a graphic on the server. Rather,

software on the server simply generates a set of instructions to other software that builds the page description file.

By building the page description file on the server without rendering the file and instead rendering the graphic for the first time on the client, the present method saves substantial computing time at the server level. The server is thus much less likely to become bogged down by simultaneous requests from multiple users, and the capacity of the server increases. Consequently, the back-end operations of the present invention are of particular importance to the efficiency of the preferred embodiment of the present invention.

Figure 9 illustrates the method of generating page description files from the perspective of the server itself. That is, Figure 9 illustrates back end operations of the present invention that make the invention more efficient than methods known in the prior art. The method begins at Step 200 when the user selects the type of document to be created, as in Step 104 of Figure 2. Information concerning the document type that the user has selected, (e.g., a business card), is sent through the Internet or other network or other type of connection to the server 12, as shown in Step 202 of Figure 9. At Step 204, the server transmits an interactive form corresponding to the selected type of document to the client. At Step 206, the interactive form is displayed at the client, and at Step 108, the user enters customization and personalization data, as in Step 108 of Figure 2.

At Step 210, the customization and personalization data is sent through the network to the server in a particular format. As discussed previously, most browsers work in the HTML language, rather than in XML. Consequently, to take advantage of aspects of XML without sacrificing the browser-compatibility of HTML, the customization and personalization data is

transmitted to the server in a manner that mimics XML. In particular, the browser transmits the user-entered data in a format that has a token, a directive and a parameter. The "token" is a standard identifier that indicates to the software on the server that an HTML statement that mimics XML is being transmitted. In the presently preferred embodiment, that "token" is "FS_DIR", although any other token can be used. A "directive" indicating the type of instruction being sent follows the "token." Standard directives in the presently preferred embodiment include: font, size, color, alignment, nudgeUp, nudgeOver, value, x, y, layer, height, and width. Various other standard directives can be created. Following the "directive" is a "value," which is the value that is to be assigned to the directive. The "value" for a font type would be the name of the font (e.g. "Helvetica"), while the "value" for a font size would be a number (e.g. "20," for a font size of 20).

As a non-limiting example, the following is a sample of HTML code that mimics XML code for purposes of the present invention. This code sample controls the font, font size and font color of an object that has been named "Company":

```
<input name="Company" size="20">

<select name="Company">

    <option value="FS_DIR:font Helvetica">Helvetica

    <option value="FS_DIR:font Helvetica-Bold">Helvetica bold

    <option value="FS_DIR:font Helvetica-Oblique">Helvetica italic
```

```
</select>

<select name="Company">

        <option value="FS_DIR:size 12">12 point

        <option value="FS_DIR:size 18">18 point

        <option value="FS_DIR:size 24">24 point

</select>

<select name="Company">

        <option value="FS_DIR:color ff0000">bright red

        <option value="FS_DIR:color 00ff00">bright green

        <option value="FS_DIR:color 0000ff">bright blue

</select>
```

In the example above, the user may choose between "Helvetica," "Helvetica Bold," and "Helvetica Italic" fonts. The user may choose either a 12, 18 or 24 point font size. The user can choose between bright red, bright green and bright blue font colors.

As seen above, standard HTML controls are used to mimic XML, in a manner that can generally be termed "pseudo-XML". In particular, the "psuedo-XML" code serves to pass an

HttpServletRequest object to the server, where the fields map to field names in the default document templates that are stored on the server.

Returning to Figure 9 and advancing to Step 212, the software on the server begins the process of compiling information about the document to be printed. Referring to Figure 1, the software in the server 12 typically comprises three main components. One component is a parser 80 which parses data statements into individual data components. A reconciler 82 reconciles the individual data components from the parser 80 into a pre-defined format for passing to a page description file builder 84. These three components 80, 82 and 84 will be described in more detail below. Server 12 also has a memory 86 in which is stored statements that define characteristics of different document templates. These characteristics include such information as default values for font size, font color, document layout, document size and various other parameters associated with the document.

The document template statements are preferably written in the XML markup language, although they may alternatively be written in various other formats known in the art. At Step 212, templates corresponding to the document type that the user has chosen are taken from memory and sent to the parser 80, which then parses the templates to extract the component parts of data stored in the template. The parser 80 also parses the pseudo XML statements that the client has transmitted to the server after the user has input customization and personalization data at the client.

The parser 80 outputs the individual data components that it has parsed from the templates in the memory 86 and from the customization and personalization data that the client 10 has transmitted to the server. The parser sends that data, which is in the form of information

packets, to a reconciler 82, as illustrated in Step 216. At Step 218, the reconciler formulates the information packets into application program interface (API) calls.

By way of background, an application program interface (API) is a set of call statements and/or a data format that an application program requires in order to be invoked. In this case, the application program is a page description file builder program. The specific application program interface (API) is different for each particular page description file generator. For example, some page description file generators require that particular data be provided in a particular format in a data file to the application program. The data that is fed to the page description file builder 84 defines the ultimate nature of the page description file that the page description file builder 84 outputs and ultimately transmits back to the client 10.

There are a variety of different page description file builder programs that can be incorporated into the present invention. Such a page description file builder may be custom-written by the developer of the system, or it may be provided as an off-the-shelf program, such as the software known as PDFlib, which is available from PDFlib GmbH of Munich, Germany. PDFlib is a portable C library that can be used on a variety of different operating system platforms. PDFlib generates the PDF code that graphically represents data that is input to PDFlib. When data is properly formatted and input to PDFlib, the PDFlib software generates PDF files of text, graphics, images, and hypertext elements, as desired. PDFlib transforms the data that is input from the reconciler 82 into a PDF file that can be viewed with such PDF file readers as Adobe® Acrobat Reader®. This step is accomplished at Step 220 of Figure 9, after which the PDF file that PDFlib generates is transmitted to the client at Step 220.

It should be noted that during the Steps 212-220, the graphic image is not rendered on the server itself. The PDF page description file is generated by the page description file builder solely on the basis of instructions that the reconciler 82 generates. Unlike certain prior art, that must render the graphic before sending the data file to the page description file builder, the preferred embodiment of the present invention never renders the graphic on the server. As a consequence, the preferred embodiment of the present invention is much faster and more efficient than prior art methods that render the graphic prior to building the page description file. Additionally, because the server has only to build instructions and does not need to perform the computationally-intensive task of rendering the graphic, the capacity of the server for receiving requests from a multiplicity of different clients, is increased. That is, because the server is less burdened than in the prior art, it is able to quickly generate page description files for a large number of users, who do not need to wait in a queue nearly as long in the preferred embodiment of the present invention in order to receive a page description file than in prior art systems.

Indeed, in the preferred embodiment of the present invention, the graphic is not rendered on the server, but is first rendered at the client in Step 220. In other words, in the preferred embodiment of the present invention, the role of the server is to generate a page description file without rendering the graphic, but merely sending the page description file to the client, which then renders the graphic for the first time. This rendering at the client end is typically done with off-the-shelf computer software such as Adobe® Acrobat Reader®. The user may then visually inspect the final document using Adobe® Acrobat Reader®, and print the file on a local or network printer with which the client is interconnected.

Alternatively, the user may save the page description file onto a floppy disk or other storage medium at the client, and then transport that storage medium to another type of printer, such as a commercial digital printer that may be located offsite or elsewhere. Alternatively, the client may transmit the data file over a network, such as an intranet or the internet, or to another device, such as a commercial printer. As a further alternative, the server may transmit the page description file somewhere other than the client. For example, the server may e-mail the page description file to an e-mail address that the user specifies on an interactive form or to some other e-mail address. In this way, the user may specify that the server should e-mail the page description file to a commercial printing facility for commercial printing. In any event, a preferred aspect of the present invention is that the step of rendering the graphic occurs locally rather than at the server, thereby greatly increasing the efficiency of the entire system.

Considering now another aspect of one embodiment of the invention, Step 212 in Figure 9 relates to reading templates that are stored in memory on the server corresponding to the chosen document. As described earlier, these templates are typically written in the XML mark-up language, and are then parsed by a data parser prior to being formatted for an application program interface (API) call or calls. In one embodiment of the invention, the templates are generated with a Graphical Use Interface (GUI). As described in Figure 10, a Graphical User Interface (GUI) is loaded into a computer, such as a personal computer, in Step 230.

In Step 232, the user uses the Graphical User Interface to define the size and the shape of the document to be designed. In Step 234, the user defines the default locations on the document of various text and graphics. In Step 236, the user uses the GUI to define default values for font size, font color, background color, and various other parameters that define the document.

Once the user has finished designing the document on the screen using the GUI, the user may save the template file as an XML file, or as any of a variety of different file formats. An example of a GUI that is suitable for use in the present invention is the GUI that is incorporated into the ProofWorks software, which is available from Avery Dennison Corporation. The GUI that is incorporated into ProofWorks can be adapted to be used as a stand alone program for designing the templates used in the preferred embodiment of the present invention. GUIs are generally well known in the art, and a variety of different off-the-shelf GUIs can be incorporated into the preferred embodiment of the present invention, although some such programs will need to be modified to save the template data in the XML format. It is generally understood in the art how to generate XML files from a GUI.

Returning to Figure 10, once the default values of such parameters as font size, font color, background color, and various other parameters are defined by the GUI, the design is saved as an XML file in Step 238. The XML file is loaded onto the server 12 (Fig. 1) and is stored in memory as a template, according to Step 240. Alternatively, the template may be stored in a database on the server, or on a long-term storage medium, such as a disk drive, or other storage medium. The template designer then exits the GUI to terminate the template creation process. Templates may include default information for a single page, or may have default information for multiple pages for multiple page documents.

Turning now to another aspect of one embodiment of the present invention, the method may be adapted to customize the page description file for the printer to which the page description file will ultimately be printed. The information about the printer to be used may be transmitted to the page description file builder, in order to optimize the output page description

file from the page description file builder for printing on the particular printer. That is, certain aspects of the output page description file may be altered when the file is to be printed on a particular printer. Some printers perform better when the page description file is formatted in a particular format, or with particular characteristics, that are particularly well-suited to that type of printer. So for example, a specific make and model of ink jet printer may perform best when the page description file includes certain statements or parameters, whereas a different type of printer, such as a laser printer, may perform more optimally when the page description file is written with certain other statements or parameters included.

One embodiment of a method of optimizing the page description file for the particular type of printer used, is described as follows. A table of printer driver properties is stored in memory on the server. Information is transmitted from the client to the server to specify the printer driver that will be used in the printing step. The server then finds the entry in the printer driver table that corresponds to the printer driver to be used, and retrieves data about the printer driver. Then, in the step of building the page description file, the page description file is constructed for compatibility with the printer driver. This feature is particularly useful in ensuring that a document has consistent alignment on the page when printed on a variety of different printers. That is, particular printers may print an image at a slight offset on the page relative to images printed by other printers. By optimizing the page description file for a particular printer type, the system can assure that different printers will print the image at a consistent location on the page.

The printer driver table is typically provided by the entity that hosts the server, based on tests that have been performed on a variety of different printers or on specifications of printer

manufacturers, and/or on known characteristics of particular printer drivers. However, as an alternative, the table may be built dynamically with data from users. For example, the users may be provided with a standard sheet printed with a grid pattern. The user may print a standard test image onto the grid pattern, and then transmit information to the server about where on the grid pattern the test image has printed. Software on the server then calculates the vertical and horizontal distances that the image has been offset on the printed page relative to the desired location on the page, and stores that offset information in the printer driver table. Then, when the system creates a page description file for the same model of printer in the future, the system can use the information stored in the printer driver table to compensate for the offset associated with that printer.

Systems according to the present invention may be incorporated into a variety of different business models. In one model, a company or individual that owns the rights to the invention (the "owner") hosts the server 12 (Fig. 1), and end-users connect with the owner-hosted server to create customized and/or personalized documents. By hosting the site itself, the owner can easily add new or modified features to the system, test new concepts, and update the software.

In a second model, the end-user connects with a web site that is hosted by an entity other than the owner. The user accesses the owner-hosted server 12 through the other's web site. This is known in the art as an Application Service Provider (ASP) model. For example, the end-user may access a special page on a major web site of a retailer that sells supplies such as blank labels or cardstock distributed by the owner. When the user selects a particular option (such as "Design a Document"), the user may be automatically rerouted to the owner-hosted web site to create the

document. After the owner-hosted server generates and delivers the page description file, the user is automatically returned to the other's web site.

Similarly, in a third model, the owner-hosted site would be accessed through "deep links" from a third party's site, such as Yahoo!®, Infoseek®, or MSN®. The owner-hosted server could be accessed from multiple pages on the third party's site, and the end-user need not be aware that he or she has been temporarily rerouted to the owner-hosted site. As with the second model, when the owner-hosted server has completed the task of generating the printable page description file of the document, the end-user is returned to the third party's site.

In a fourth model, the owner provides software and perhaps default document templates to a third party that then loads the software onto a third party-hosted site. The owner would typically provide the third party with updates to the software, as necessary. The owner would generally provide the software to the third party under a license, but could alternatively sell the software to the third party. The owner may also provide the software to be used as a component in a "suite" of different tools that perform various functions. Various other business models can be imagined.

It should be noted that the owner in these various models may be a company that manufactures, sells and/or distributes printable media, such as blank labels, perforated sheets of blank business cards, and/or a wide variety of other printable media. The owner could then offer its site as a free service to customers that have purchased the owner's printable media products. Alternatively, the owner may be a company that does not manufacture or sell printable media, but that generates revenue from the site with banner advertising and/or by charging a fee to the end-user. Various other ownership scenarios can be imagined.

In all of the models described above, the ultimate destination of the printable page description file need not be to the client or to a client-controlled printer. As explained previously, the server may transmit the final printable page description file to the client for printing on a client-controlled printer. However, the server may alternatively send the printable page description file to a location other than the client, such as to a commercial printing facility and/or a printing press. The server may alternatively send the page description file via e-mail to an e-mail address or via FTP (File Transfer Protocol) to an FTP address. The user may specify the e-mail or FTP address to which the file should be sent, or the e-mail or FTP address may be stored in a directory on the server, such as a directory of commercial printing facilities.

The foregoing has described various preferred aspects of embodiments of the present invention. However, it should be understood that the present invention may be varied in different ways, and still fall within the broad outline of the present invention. That is, various changes and modifications may be made to the previously described embodiments and still be within the broad outline of the invention. For example, the presently preferred embodiment of a page description file is a PDF file, which is consistent with a format developed by Adobe Systems Incorporated. However, the present invention is not limited to using PDF files. Other types of page description files known in the art or to be developed in the future may be used. The PDF file format is presently preferred because the Adobe® Acrobat Reader® software is so widely distributed among personal computer users. However, it is expected that over time, other high quality page description file readers will become widely distributed. It is even possible to incorporate custom-written software in the form of a plug-in to the browser on the client 10 for reading the page description files that the server 12 generates and transmits back to the client 10.

Consequently, the particular type of graphic reading and rendering software that is used on the client is not a limitation of the present invention.

Another modification to the present invention relates to using actual XML code with the browser rather than the pseudo-XML code that is described above. At present, pseudo-XML code is used because many of the browsers that are in wide distribution among personal computer users do not understand true XML. Instead, these older browsers understand only the HTML mark-up language, which is different than XML. Consequently, the motivation for using pseudo-XML is largely due to the limited number of browsers that can communicate in true XML. However, it is possible that over time, the majority of browsers in use will be XML compatible. At that time, it is expected that the user-input data from the client will be transmitted in true XML format back to the server 12.

Another modification that may be made relates to the ultimate printer that will print the page description file that the server has generated. In the presently preferred embodiment, the ultimate printer is a typical office or home printer such as an inkjet or laser printer. Of course, various other printers may be used, such as digital printers and/or color laser printers. However, the present invention is not limited to printing the page description file on a typical office or home printer. For example, the user may prefer to print the page description file with high end commercial digital printers. In that case, the page description file will typically be transferred from the client 10 to another computer or device via a network, for example, or by simply transferring the page description file to the commercial printer on a hard storage media, such as a floppy disk or Zip® disk. In other words, the present invention is not limited to printing the page

description file on a laser or inkjet printer, but also encompasses printing the page description file in a commercial printing facility or on various high end printers.

A further modification is for the system to generate high-resolution page description files, but never actually print them on a printer. The page description file may be intended for ultimate display on a monitor, an electronic book, or a wide variety of other devices. Alternatively, the end-user may have the option of printing the page description file, or choosing not to print the file and merely displaying it on a display device.

The present invention may be adapted to generate page description files for printing on a wide variety of media. The present invention is not limited to any particular media, but can be used in conjunction with any printable media. For example, in addition to symmetrical or asymmetrical paper-based media, the files may be printed on transparencies and fabrics, films, image transfer media such as hot or cold-peel transfer media, various types of labels, large-format media, photo-quality media, various adhesively-backed media such as labels and stamps, laminated media, and many other printable types of media. Additionally, there are innumerable applications for the technology, such as for printing signage, tickets, checks, forms, postage stamps, calendars, personal planners, photographs, identification cards, bar-coded and/or cryptographic media, puzzles, two-dimensional media that is foldable into three-dimensional media, and many other applications, including printing very large-sized media with special printers.

It is also important to note that the use of interactive forms is only one of many ways to provide the server with information needed to generate a page description file. For example, personalization and/or customization information may be transmitted to the server from another

server. In one alternative embodiment, the user accesses a web page from an Internet site. Based on information that the user transmits to the site, and/or on user information already stored on the site, or on information that the site itself generates, or on information that the site obtains from elsewhere, or a combination of any of the foregoing, a server associated with the site may transmit information to the server of the present system for formatting into a customized and/or personalized document. Similarly, name and address information that is stored online may be sent from the online storage site to the server for processing into personalized and/or customized labels according to the present invention.

Figure 11 represents a method in which customization and/or personalization information is not transmitted from the end user to the server, but is transferred from a first server to a second server. At Step 260, a first server transmits customization and/or personalization information to the second server. The information may be pre-stored on the first server as, for example, information that the user has previously stored on the first server, or information that the first server has itself generated. The second server then generates a page description file at Step 262, based on the customization and/or personalization information received from the first server. The second server transmits the page description file to the client at Step 264, and the page description is graphically rendered for the first time, at the client, in Step 266. The rendered document may then be printed at a client-controlled printer in Step 268.

The present invention is not limited to client-server network models, but also extends to client-to-client models and, as explained above, to server-to-server models. In a client-to-client model, a user might enter customization and/or personalization data at one client and transmit that data to a second client, which then generates a page description file. The page description

file is then transmitted from the second client to the first client for rendering, or rendered on the second client itself. Alternatively, the second client may transmit the page description file elsewhere, such as to a printer or other device or to another computer.

Consequently, the present invention is not limited to systems in which a user enters the customization and/or personalization information at the client and then transmits that information from the client to the server. The information transmitted to the server may even be generated in real time as, for example, data from a testing device, real-time sports scores, weather data, or an almost infinite variety of different types of data that can be represented graphically in some fashion. As a further alternative, the data provided to the server may be data files or data drawn from data files, such as data files for personal organization software or a variety of other programs that may be resident on a client or server. The data may also be stored in and drawn from a database.

The present invention is also not limited to generating page description files on the server, but also encompasses generating bit-mapped graphic files on the server and transmitting them elsewhere. However, the graphic file is preferably not rendered on the server, but is first rendered at the client or at another site after the server has generated the file. In this manner, the server does not become engaged in the resource-intensive process of rendering a graphic file. Software for building graphic files is known in the art.

Consequently, the present invention is not limited to the preferred embodiments described above.